Reflection

SoftUni Team

Technical Trainers

Software University

http://softuni.bg

Table of Contents

SoftUni Foundation

Questions

SoftUni Foundation

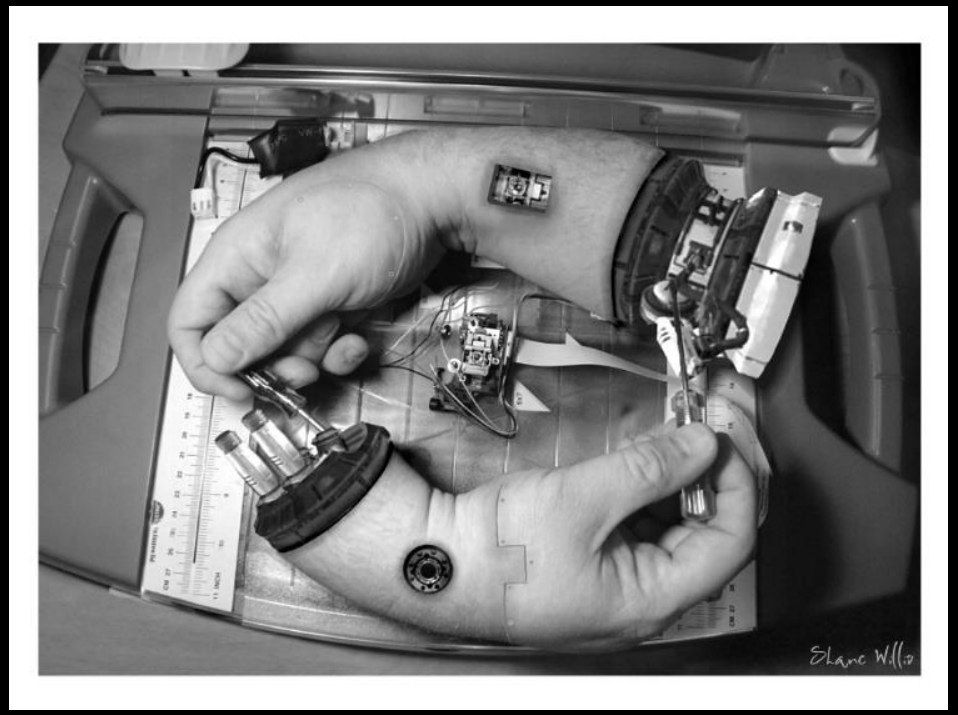
sli.do

#JavaFundamentals

What is Metaprogramming?

SoftUni Foundation

- **Programming technique** in which computer programs have the ability to treat **programs as their data**
- Program can be designed to:
  - **Read**
  - **Generate**
  - **Analyze**
  - **Transform**
- **Modify itself** while **running**.

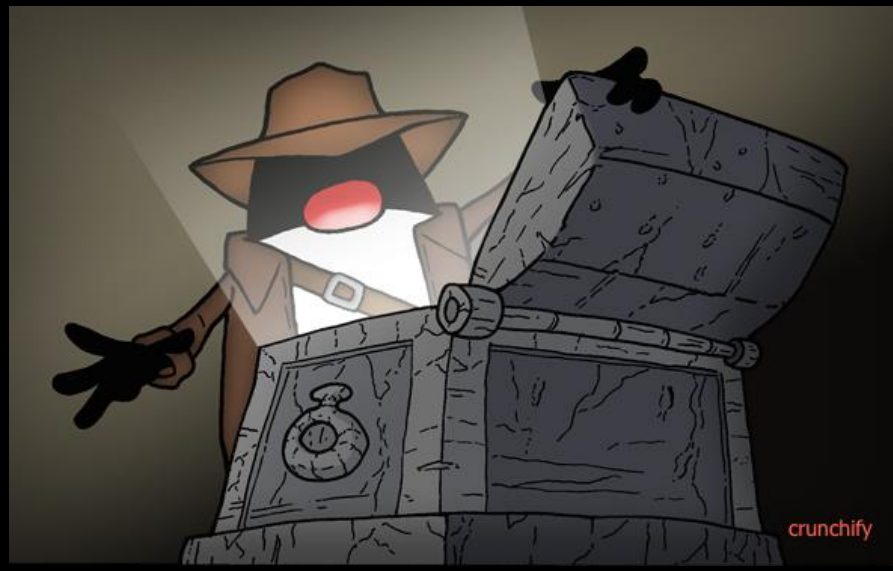


What is Reflection?

SoftUni Foundation

"In computer science, **reflection** is the ability of a computer program to **examine**, **introspect**, and **modify** its own structure and behavior at **runtime**."

- **Extensibility features**
- **Class libraries** and visual **development environments**
- **Debuggers** and **test tools**



What is Reflection?

- If it is possible to perform an operation **without using reflection**, then it's preferable to **avoid using it**

- **Performance Overhead**

- **Security Restrictions**

- **Exposure of Internals**

The Class Object

SoftUni Foundation

- Obtain its `java.lang.Class` object
  - If you **know** the **name**

```
Class myObjectClass = MyObject.class
```

  - If you **don't** know the name at **compile time**

```
Class class = Class.forName(className);
```

You need fully qualified class name as String

Class Name

SoftUni Foundation

- Obtain `Class` name
  - Fully qualified class name

```
String className = aClass.getName();
```

  - Class name without the package name

```
String simpleClassName = aClass.getSimpleName();
```

Base Class and Interfaces

SoftUni Foundation

- Obtain **parent class**

```
Class className = aClass.getSuperclass();
```

- Obtain **interfaces**

```
Class[] interfaces = aClass.getInterfaces();
```

  - **Interfaces** are also **represented** by `Class` objects in Java Reflection
  - Only the interfaces **specifically declared** implemented by a given class are **returned**

Problem: Reflection

SoftUni Foundation

- Import ReflectionClass to your **src** folder in your project
- Using reflection print:
  - This class type
  - Super class type
  - All Interfaces
  - Instantiate object using reflection and print it
- Don't change anything in class

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

Solution: Reflection

SoftUni Foundation

```
Class aClass = Reflection.class;
System.out.println(aClass);
System.out.println(aClass.getSuperclass());
Class[] interfaces = aClass.getInterfaces();
for (Class anInterface : interfaces) {
  System.out.println(anInterface);
}
Reflection ref = (Reflection) aClass.newInstance();
System.out.println(ref);
```

Create new object

You must cast

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

SoftUni Foundation

Reflection

Live Exercises in Class (Lab)

Constructors

SoftUni Foundation

- Obtain **only public constructors**

```
Constructor[] ctors = aClass.getConstructors();
```

- Obtain **all constructors**

```
Constructor[] ctors =
              aClass.getDeclaredConstructors();
```

- Get constructor by **parameters**

```
Constructor ctor =
            aClass.getConstructor(String.class);
```

Constructors (2)

SoftUni Foundation

- Get **parameter types**

```
Class[] parameterTypes =
                    constructor.getParameterTypes();
```

- **Instantiating objects** using constructor

```
Constructor constructor =
MyObject.class.getConstructor(String.class);
MyObject myObject = (MyObject)
constructor.newInstance("arg1", "arg2"...);
```

Fields Name and Type

SoftUni Foundation

- Obtain **public** fields

```
Field field = aClass.getField("somefield");
Field[] fields = aClass.getFields();
```

- Obtain **all** fields

```
Field[] fields = aClass.getDeclaredFields();
```

- Get field **name and type**

```
String fieldName = field.getName();
Object fieldType = field.getType();
```

Fields Set and Get

SoftUni Foundation

- Setting value for field

```
Class aClass = MyObject.class
Field field = aClass.getField("someField");
MyObject objectInstance = new MyObject();
Object value = field.get(objectInstance);

field.set(objetInstance, value);
```

The objectInstance parameter passed to the get and set method should be an instance of the class that owns the field

Methods

- Obtain **public** methods

```
Method[] methods = aClass.getMethods();
Method method =
    aClass.getMethod("doSomething",String.clas);
```

- Get methods without **parameters**

```
Method method =
          aClass.getMethod("doSomething", null);
```

Method Invoke

- Obtain method **parameters** and **return type**

```
Class[] paramTypes = method.getParameterTypes();
Class returnType = method.getReturnType();
```

- Get methods without **parameters**

```
Method method =
 MyObject.class.getMethod("doSomething", String.class);
Object returnValue = method.invoke(null,
"arg1");
```

null is for static methods

Problem: Getters and Setters

SoftUni Foundation

- Using **reflection** get all methods and print:
- **Sort** getters and setters **alphabetically**
- **Getters**:
  - A getter method have its name start with "get", take 0 parameters, and returns a value.
- **Setters**:
  - A setter method have its name start with "set", and takes 1 parameter

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

Solution: Getters

SoftUni Foundation

```
Method[] methods = aClass.getDeclaredMethods();
List<Method> getters = new ArrayList<>();
for (Method method : methods) {
  if (method.getName().startsWith("get")) {
    if (method.getParameterTypes().length == 0) {
      getters.add(method);
    }
  } }
//TODO: Print getters sorted alphabetically
```

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

Solution: Setters

SoftUni Foundation

```
List<Method> setters = new ArrayList<>();
for (Method method : methods) {
  if (method.getName().startsWith("set")) {
    if (method.getParameterTypes().length == 1) {
      if (void.class.equals(method.getReturnType())) {
        setters.add(method);
      }
    } } }
//TODO: Print setters sorted alphabetically
```

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

SoftUni Foundation

Constructors, Fields and Methods

Live Exercises in Class (Lab)

Access Modifiers

SoftUni Foundation

- Obtain the **class modifiers** like this

```
int modifiers = aClass.getModifiers();
```

- Each modifier is a **flag bit** that is either set or cleared
- You can check the modifiers

getModifiers() can be called on constructors, fields, methods

```
Modifier.isPrivate(int modifiers)
Modifier.isProtected(int modifiers)
Modifier.isPublic(int modifiers)
Modifier.isStatic(int modifiers)
```

Annotations

SoftUni Foundation

- Obtain class annotations

```
Annotation[] annotations = aClass.getAnnotations();
Annotation annotation = aClass.getAnnotation(MyAnno.class);
```

- Obtain parameter annotations

```
Annotation[][] parameterAnnotations =
                                    method.getParameterAnnotations();
```

- Obtain fields and methods annotations

```
Annotation[] fieldAnots = field.getDeclaredAnnotations();
Annotation[] methodAnot = method.getDeclaredAnnotations();
```

Arrays

SoftUni Foundation

- Creating arrays via Java Reflection

```
int[] intArray = (int[]) Array.newInstance(int.class, 3);
```

- Obtain parameter annotations

```
Array.set(intArray, 0, 123);
Array.set(intArray, 1, 456);
```

- Obtain fields and methods annotations

```
Class stringArrayComponentType =
                        stringArrayClass.getComponentType();
```

Problem: High Quality Mistakes

SoftUni Foundation

- You perfectly know how to write High Quality Code
- Check **Reflection class** and print all mistakes in **access modifiers** which you can find
- Get all fields, getters and setters and sort each category by name
- First print mistakes in **fields**
- Then print mistakes in **getters**
- Then print mistakes in **setters**

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

Solution: High Quality Mistakes

SoftUni Foundation

```
List<Field> fields = Arrays.asList(aClass.getDeclaredFields());
fields.sort(new Comparator<Field>() {
  @Override
  public int compare(Field o1, Field o2) {
    return o1.getName().compareTo(o2.getName());
  }
});
```

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

Solution: High Quality Mistakes(2)

SoftUni Foundation

```
for (Field field : fields) {
  if (!Modifier.isPrivate(field.getModifiers())) {
    System.out.println(field.getName() + " must be private!");
  }
}
```

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

Solution: High Quality Mistakes(3)

SoftUni Foundation

```
List<Method> methods =
      Arrays.asList(aClass.getDeclaredMethods());
sort(methods);
for (Method method : methods) {
  if (method.getName().startsWith("get")) {
    if (method.getParameterTypes().length == 0) {
      if (!Modifier.isPublic(method.getModifiers())) {
        System.out.println(method.getName() +
                                    " have to be public!");
      }}}}                          //TODO: do the same for setters
```

Check your solution here: https://judge.softuni.bg/Contests/Compete/Index/526#0

Summary

- What is Reflection
- Reflection API
  - Reflecting Classes
  - Reflecting Constructors
  - Reflecting Fields
  - Reflecting Methods
  - Reflecting Annotations
  - Access Modifiers

Reflection

SoftUni Foundation

Questions?

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

Software University

SoftUni Foundation

FORUM

License

SoftUni Foundation

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license

- Attribution: this work may contain portions from
  - "Fundamentals of Computer Programming with Java" book by Svetlin Nakov & Co. under CC-BY-SA license
  - "OOP" course by Telerik Academy under CC-BY-NC-SA license